4П

УДК 681.223

І.А. Пилипенко
Івано-Франківський інститут менеджменту та економіки «Галицька академія», Україна
inna_mail@yahoo.com

Архіватори знань про одновимірні та двовимірні об’єкти діагностування

У даній статті запропоновані методи стиснення одновимірних (текстових) та двовимірних (графічних) даних на основі базису Галуа, які дозволяють в деякій мірі зменшити об’єм інформації, що зберігається, обробляється або передається, мінімізувати втрату інформації при відновленні.

Архіватори – це програми для створення архівів. Архіви призначені для збереження даних у зручному компактному виді. У якості даних звичайно виступають файли і папки. Як правило, дані попередньо піддаються процедурі стиснення або упакування. Тому майже кожен архіватор одночасно є програмою для стиснення даних. З іншого боку, будь-яка програма для стиснення даних може розглядатися як архіватор. Ефективність стиснення є найважливішою характеристикою архіваторів. Від неї залежить розмір створюваних архівів. Чим менше архів, тим менше місця потрібно для його збереження. Для передачі потрібна менша пропускна здатність каналу передачі або затрачується менший час. Переваги архівів очевидні, якщо врахувати, що дані зменшуються в розмірі й у 2 рази, і в 5 разів.

Стиснення даних використовується дуже широко. Можна сказати, майже скрізь. Наприклад, документи PDF, як правило, містять стиснуту інформацію. Досить багато виконуваних файлів EXE стиснуті спеціальними пакувальниками. Різні мультимедійні файли (GIF, JPG, MP3, MPG) є своєрідними архівами.

Основним недоліком архівів є неможливість прямого доступу до даних. Їх спочатку необхідно витягти з архіву або розпакувати. Операція розпакування, утім, як і упакування, вимагає деяких системних ресурсів. Це не миттєва операція. Тому архіви в основному застосовують з порівняно рідко використовуваними даними. Наприклад, для збереження резервних копій або інсталяційних файлів.

У найпростішому випадку архіватор дозволяє тільки упакувати або розпакувати один файл. Крім власне стиснення даних, сучасні архіватори забезпечують деякі додаткові функції. Можна виділити декілька основних:

- стиснення деяких файлів і цілих директорій;
- створення архівів, що саморозпаковуються (SFX). Тобто для розпакування архіву програму-архіватор не потрібно;
- зміна вмісту архіву;
- шифрування вмісту архіву;
- інформація для відновлення архіву при частковому ушкодженні і можливість відновлення ушкоджених архівів;
- розбиття архіву на декілька частин або томів;
- консольна версія програми для роботи з командного рядка;
- графічна (GUI) версія програми.

Слід зазначити, що дані після «стиснення» архіваторами набувають специфічного значення, яке «розуміє» тільки сам архіватор, особливо якщо метод стиснення унікальний. Інші програми таких даних використовувати не можуть.

Стиснення буває без втрат (коли можливо відновлення вихідних даних без спотворень) або з втратами (відновлення можливе з незначними спотвореннями). Стиснення без втрат використовується при обробці та збереженні комп'ютерних програм і даних. Стиснення з втратами зазвичай застосовується для зменшення об'єму звукової, фото- та відеоінформації. І, як показує практика, стиснення з втратами для такого роду інформації є набагато вигіднішим.

Стиснення текстових даних базується на усуненні надлишку інформації, яка міститься у вихідних даних. Прикладом надлишку є повторення в тексті фрагментів (наприклад, слів природної або машинної мови). Подібний надлишок зазвичай усувається заміною повторюваних послідовностей більш коротким значенням (кодом). Інший вид збитковості пов'язаний з тим, що деякі значення в даних, що стискаються, зустрічаються частіше інших, при цьому можна замінювати дані, що часто зустрічаються, більш короткими кодами, а ті, що рідко, більш довгими (імовірнісне стиснення). Стиснення даних, які не мають властивості надлишку (наприклад, випадковий сигнал чи шум), неможливе. Так само зазвичай неможливо стиснути зашифровану інформацію [1].

1. Метод одномірного кодування текстових даних в базисі Галуа

Коди Галуа утворюються згідно з наступною формулою:

$$G_{i+1} = G_i \oplus G_{i-n}.$$

Приклад коду Галуа з об'ємом коду V = N.

1	2	3	4	5	6	7	8
0	0	0	1	0	1	1	1

Використання восьмибітового слова дає змогу закодувати $2^8 = 256$ знаків, тоді як реальні алфавіти з урахуванням цифр і деяких допоміжних символів містять до 50 – 60 знаків, тобто для кодування їх потрібні п'яти-шестибітові комбінації та аналогічні структури пам'яті.

Спробуємо зменшити кількість біт, необхідних для кодування одного символу, поділивши всі символи на чотири групи, що показані в табл. 1.

Таблиця 1 – Кодова таблиця Галуа з поділом на групи

1	1	1	1	1	0	0	1	1	0	1	0	0	1	0	0	0	0	1	0	1	0	1	1	1	0	1	1	0	0	0	1
				x	t	h	e	s	i	n	d	o	r	a	m	p	l	y	c	u	b	j	f	g	w	k	v	q	z	.	_
				н	к	а	й	м	о	в	і	р	н	я	с	т	у	е	д	о	ц	л	г	ш	з	х	п	ь	ч	и	_
				б	ф	щ	ю	б	ж	ъ	ы	э	є	ї	ё	ч	0	1	2	3	4	5	6	7	8	9	+	-	=		_
				,	;	:	?	!	“	№	%	*	(	)	~	`	“	#	$	^	&	|	\	/	<	>	{	}	[	]	_

Суть методу полягає в наступному. Замість кожного символу тексту записуються відповідні 5 біт коду з таблиці. Якщо наступний символ тексту відповідає наступному символу кодової таблиці, то записується тільки 1 біт, в іншому випадку – інвертований біт Галуа і 5 біт, що відповідають заданому символу.

Останні три біти послідовності Галуа визначають номер групи, в якій знаходиться заданий символ: 00011 – друга група; 00111 – третя група; 01111 – четверта група.

Якщо ми знаходимось у певній групі, а хочемо закодувати символ латинського алфавіту, потрібно ще раз вказати код заданої групи. Для кодування великої літери будемо використовувати послідовність 00000.

Спробуємо даний метод для кодування символів з різних груп. Так, слово Windows98 буде зашифровано наступним чином:

W	$\overline{G}$	i	n	d	o	$\overline{G}$	w	$\overline{G}$	s	$\overline{G}$	III група	9	$\overline{G}$	8
00000 01110	0	00110	1	0	0	0	01110	0	10011	1	00111	11101	0	01110

Дане слово займає 72 біт пам'яті. Після упаковки його за допомогою 5-бітової послідовності Галуа воно займатиме 48 біт. Отже, Кс = 1,5.

Коефіцієнт стиснення даних буде розраховуватися за формулою:

$$Kc = \frac{8 \cdot n}{(6 \cdot n - 5 \cdot n_p + 5 \cdot n_v + 5 \cdot n_r)}, \tag{1}$$

де n – кількість символів в файлі, що кодується, n_v – кількість великих символів, n_r – кількість переходів з однієї групи кодової таблиці Галуа в іншу, де n_p – кількість символів, які в таблиці Галуа знаходяться поруч [2].

2. Стиснення графічних зображень

Комп'ютерна графіка традиційно поділяється на два види: векторну та растрову. Векторне зображення складається з набору відрізків, багатокутників, кривих, що задані в деякій системі координат і описані математично. З векторними даними пов'язана інформація про їхні атрибути (наприклад, колір, товщина, тощо). Векторне зображення легко масштабувати, повертати, нахиляти. Тому воно не залежить від пристрою відображення – чи то монітор з роздільною здатністю 72 dpi, чи принтер з роздільною здатністю 600 dpi. Файли векторного формату корисні для збереження лінійних елементів (наприклад, ліній та прямокутників), а також елементів, які можна розкласти на прості геометричні об'єкти (наприклад, текст). Растрове зображення – це масив цифрових значень, що визначають колір окремих пікселів. Кількість бітів на піксель визначає кількість кольорів, що можуть задаватися для цього пікселя. Файли цього типу добре підходять для збереження реальних зображень, наприклад, фотографій та відеозображень. Існує ще третій тип – метафайли – це файли, де міститься як векторна частина зображення, так і растрова.

Але зображення майже ніколи не записується у файл просто так. Як правило, воно кодується за деяким алгоритмом, крім того спочатку додається блок службової інформації (наприклад: ідентифікатор формату, версія формату, роздільна здатність по горизонталі, вертикалі, кількість бітів на колір, тип кодування, кількість площин, палітра, кількість пікселів на дюйм, тощо). Найчастіше графічні дані кодуються з метою зменшення об'єму файла, але можлива й інша мета (наприклад, щоб: файли не могли продивитися сторонні люди, визначити, чи не спотворені дані (наприклад, за методом CRC), тощо). Кодування бувають таких типів: упаковування пікселів, RLE, LZW, Huffmen або CCITT, DCT або JPEG, JBID, ART, Fractal, або будь-якого, який ви самі можете придумати.

За іншою класифікацією графічні файли поділяються на власні (що належать деякій графічній програмі) та обміну (що створені для обміну між декількома програмами). Майже кожна графічна програма має свій власний формат, а для обміну даними з іншими програмами використовує формати обміну.

Зрозуміло, що кожен формат був створений розробником з певною конкретною метою. Ми розглянемо декілька сфер застосування графічних файлів та формати, які доцільно використовувати в кожному випадку.

В Інтернеті найпоширенішими форматами є GIF та JPEG. Головним принципом, за яким створювалися ці формати, є мінімізація розміру файлу для передачі у мережі з низькою пропускною здатністю. GIF підтримує палітрові дані з максимальною кількістю кольорів 256 (без прозорості), анімацію, хоча остання модифікація може мати у кожному кадрі свою палітру і при використанні прозорих областей можна досягти більшої кількості кольорів. У випадку, коли у вихідному форматі було до 256 кольорів, перетворення на GIF відбудеться без втрат, а якщо ні, то потрібно виконати тонування або визначити кольори, що найчастіше використовуються, і подібні кольори перетворити в один, щоб загальна кількість кольорів не перевищувала 256. JPEG є у деякому плані альтернативою GIF, тому що підтримує 24 біти на колір, але аналогічно до GIF, зони з схожими кольорами перетворюються на один колір, причому навіть при низькому ступені стиснення втрати неминучі. Останнім часом був створений формат PNG, що на відміну від GIF підтримує повну гаму кольорів (до 48 біт) і використовує алгоритм запаковки, схожий на LZW. Але він поки що не настільки поширений, як два попередніх.

Серед векторних форматів для Інтернету слід назвати Flash та VRML. VRML – це формат тривимірних сцен, для перегляду яких потрібен додатковий модуль – VRML plug-in (наприклад, Cosmo Player). Цей формат підтримує декілька базових геометричних фігур, різні типи освітлення, матеріалів, текстур, шрифтів. Flash – це унікальний метафайловий формат, що підтримує анімацію, морфінг, градієнтну прозорість, шари, звук, реакцію на події від мишки чи клавіатури, тощо. Для його перегляду потрібен інший plug-in – Shockwave Flash, а для створення існує лише єдиний редактор – Macromedia Flash, який є досить простим векторним редактором з підтримкою лінії часу (timeline), але можливостей якого досить для написання складних сторінок.

Без сумніву, при роботі з конкретною програмою результат роботи варто зберігати у її рідному форматі, що підтримує, як правило, усі можливості та нюанси цієї програми. Але для обміну файлів (наприклад, з видавництвом) варто використати формати TIFF для растрових даних та EPS для векторних чи метафайлових. При цьому не будуть виникати проблеми з підтримкою цих форматів у іншій версії чи на іншій платформі. Формат TIFF має два варіанти: для PC та для Mac. Починаючи з версії 5.0 (1988) він підтримує запаковку LZW, а з 6.0 (1992) – JPEG. На сьогодні це один з найуніверсальніших форматів, що підтримує кольори від 1 до 24 бітів, стиснення за методами RLE, LZW, CCITT 3, CCITT 4, JPEG, без стиснення, декілька зображень у одному файлі, альфа-канали. Формат EPS є також дуже універсальним – він підтримує запис даних у текстовому та двійковому форматі, зображення у монохромному, напівтоновому та кольоровому режимі, DCS, роздільну здатність від 75 до 3000 dpi, мініатюри (preview) та шляхи обрізки (clipping path).

У електронних презентаціях немає якого-небудь стандартного формату графічних файлів, кожен виробник підтримує формати, які він вважає за потрібне. Багато з виробників створюють свої формати. Більшість програми створення електронних презентацій (наприклад, Microsoft PowerPoint, Lotus Smart Suite, Corel Presentation, Astound) підтримують декілька (до 20) різних графічних форматів [3].

3. Метод двомірного кодування зображень в базисі Галуа

Розглянемо можливість застосування кодів Галуа для стиснення двовимірних об'єктів, а саме зображень. Для цього представимо графічне зображення у вигляді сітки (рис. 1), кожна комірка якої відповідає певному біту послідовності Галуа.

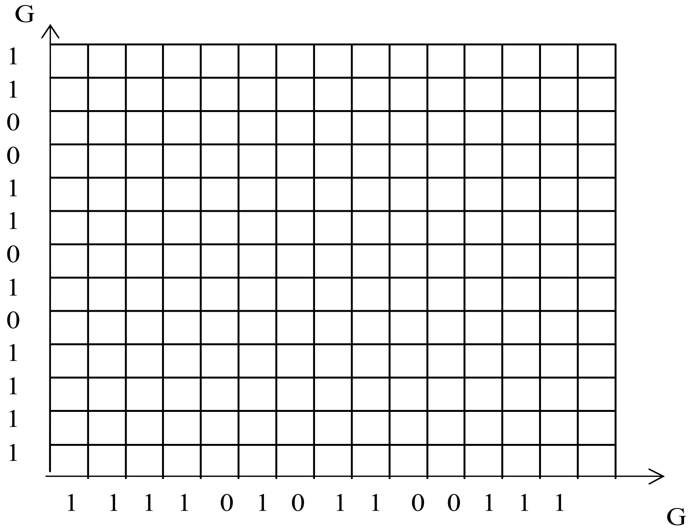


Рисунок 1 – Приклад сітки за послідовністю Галуа

Таким чином ми отримаємо матрицю, що відповідає заданій сітці:

11 11 11 11 10 11 10 11 11 10 10 11 11 11
11 11 11 11 10 11 10 11 11 10 10 11 11 11
01 01 01 01 00 01 00 01 01 00 00 01 01 01
01 01 01 01 00 01 00 01 01 00 00 01 01 01
11 11 11 11 10 11 10 11 11 10 10 11 11 11
11 11 11 11 10 11 10 11 11 10 10 11 11 11
01 01 01 01 00 01 00 01 01 00 00 01 01 01
11 11 11 11 10 11 10 11 11 10 10 11 11 11
01 01 01 01 00 01 00 01 01 00 00 01 01 01
11 11 11 11 10 11 10 11 11 10 10 11 11 11
11 11 11 11 10 11 10 11 11 10 10 11 11 11
11 11 11 11 10 11 10 11 11 10 10 11 11 11
11 11 11 11 10 11 10 11 11 10 10 11 11 11

Для кодування рисунка можна скористатися кодовою таблицею (табл. 1). Тільки в даному випадку ми будемо замість тексту використовувати кольори. Нехай перший рядок таблиці буде відповідати червоному кольору, другий – зеленому, а третій – синьому.

Починати кодування будемо з вказання біта Галуа, що відповідає першому рядочку і першому стовпчику. Якщо в заданому місці колір відсутній, то перший біт інвертуємо. В іншому випадку записуємо по п’ять бітів з таблиці, що відповідають червоному, зеленому та синьому кольорам і переходимо до наступної комірки. Якщо наступний колір повністю співпадає з попереднім, то біт, який відповідає номеру стовпчика, інвертуємо. Інакше – переходимо до кодової таблиці. Порівнюємо кожен колір з попереднім, при їх співпадінні інвертуємо наступний біт, в іншому випадку записуємо біти, що відповідають кольору. При переході на наступний рядок інвертуємо останні біти, що відповідають номеру рядочка та стовпчика.

Розглянемо наступний приклад. Нехай заданий рисунок з наступними координатами кольорів (табл. 2).

Таблиця 2 – Координати кольорів

	R	G	B	R	G	B	R	G	B	R	G	B	R	G	B
1	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
0	255	255	255	148	56	148	148	56	148	54	56	148	54	56	148
1	255	255	255	47	56	145	47	56	145	255	255	255	255	255	255
1	255	255	255	122	158	59	54	123	145	78	28	145	78	28	145
1	255	255	255	123	150	48	123	150	48	128	150	59	255	255	255
G		1			1			1			0			1	

Після кодування його за допомогою бітів Галуа отримаємо (табл. 3):

Таблиця 3 – Кодування рисунка послідовністю Галуа

01	01			01			00			01
11	11GGGGG	$\overline{G}$	GGGGG	00			11GGGGG	$\overline{G}$	$\overline{G}$	0010
01	11GGGGG	GGGGG	GGGGG	10			01			0100
01	11GGGGG	GGGGG	GGGGG	11GGGGG	GGGGG	GGGGG	10GGGGG	GGGGG	GGGGG	1000
01	11GGGGG	GGGGG	GGGGG	01			10GGGGG	$\overline{G}$	GGGGG	0100

Де G – біт послідовності Галуа, що відповідає заданому символу, а $\overline{G}$ – інвертований біт Галуа.

Відсутність кольору або його повторення кодується двома бітами, нові кольори – 17 бітами. При збереженні одного кольору і зміні іншого, не змінений колір буде позначатися одним бітом.

Отже, як видно з табл. 2, початковий рисунок має розмір 600 біт, після кодування його за допомогою послідовності Галуа його розмір становитиме 162 біти. Таким чином, розмір рисунку буде зменшено в 3,7 раз. Якщо кольори кодувати 8-бітовими послідовностями, то розмір складатиме 182 біти, а коефіцієнт стиснення становитиме 2,7.

Висновки

У роботі описаний метод стиснення алфавітно-цифрових даних за допомогою 5-бітової послідовності Галуа. Наведено формулу розрахунку коефіцієнта стиснення, що залежить від кількості великих символів, кількості переходів з однієї групи кодової таблиці Галуа в іншу та кількості символів, які в таблиці Галуа знаходяться поруч.

Представлені методи стиснення графічних даних. Проведено модифікацію рекурентного методу стиснення текстових даних із застосуванням його для графічних файлів.

Як видно з прикладів, даний метод є більш ефективний для одноалфавітних текстових даних з мінімальною кількістю великих літер та максимальною кількістю символів, розташованих поруч в кодовій таблиці. А також для графічних файлів, що мають великі області однакового або білого кольору.

Література

1. Экслер А.Б. Архиваторы (Программы для хранения и обработки информации в сжатом виде). – М: Малое предприятие «Алекс», 1992. – 150 с.
2. Pylypenko I. Theoretical basis and examples of the use of Galua codes for the data compression // Proc. of the International Conf. ACSN'2007. September 20-22, 2007. – Lviv, Ukraine. – 2007. – P. 168-169.
3. Мюррей Д., Райпер У. Энциклопедия форматов графических файлов. – К.: BHV, 1997.

И.А. Пилипенко

Архиваторы знаний об одномерных и двумерных объектах диагностирования

В данной статье предложены методы сжатия одномерных (текстовых) и двухмерных (графичных) данных на основе базиса Галуа, которые позволяют в некоторой степени уменьшить объем информации, которая сохраняется, обрабатывается или передается, минимизировать потерю информации при восстановлении.

I.A. Pylypenko

Knowledge Archivers on One-dimensional and Two-dimensional Objects of Diagnosis

In the given article the methods of text and graphical data compression on the basis of base Galoua are offered, which allow to decrease the volume of information, that is kept, processed or passed, to minimize the loss of information at renewal.

Стаття надійшла до редакції 09.07.2008.